Context-Attentive Embeddings for Improved Sentence Representations

Douwe Kiela[†], Changhan Wang[†] and Kyunghyun Cho[†‡]
[†] Facebook AI Research; [‡] New York University
{dkiela,changhan,kyunghyuncho}@fb.com

Abstract

While one of the first steps in many NLP systems is selecting what embeddings to use, we argue that such a step is better left for neural networks to figure out by themselves. To that end, we introduce a novel, straightforward yet highly effective method for combining multiple types of word embeddings in a single model, leading to state-of-the-art performance within the same model class on a variety of tasks. We subsequently show how the technique can be used to shed new insight into the usage of word embeddings in NLP systems.

1 Introduction

It is no exaggeration to say that word embeddings have revolutionized NLP. From early distributional semantic models (Turney and Pantel, 2010; Erk, 2012; Clark, 2015) to deep learning-based word embeddings (Bengio et al., 2003; Collobert and Weston, 2008; Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2016), word-level meaning representations have found applications in a wide variety of core NLP tasks, to the extent that they are now ubiquitous in the field (Goldberg, 2016).

A sprawling literature has emerged about what types of embeddings are most useful for which tasks. For instance, there has been extensive work on understanding what word embeddings learn (Levy and Goldberg, 2014b), evaluating their performance (Milajevs et al., 2014; Schnabel et al., 2015; Bakarov, 2017), specializing them for certain tasks (Maas et al., 2011; Faruqui et al., 2014; Kiela et al., 2015; Mrkšić et al., 2016; Vulić and Mrkšić, 2017) and learning sub-word level representations (Wieting et al., 2016; Bojanowski et al., 2016; Lee et al., 2016).

That said, word embeddings do not come without their problems: they are hard to evaluate, their usefulness for downstream tasks is hard to predict and they do not take into account contextual information when used (or evaluated). In this work, we address some of the weaknesses of word embeddings by making them context-attentive: we learn to represent word meaning as a weighted combination of a multitude of different word embeddings, where the weighting depends on the context.

While one of the first steps in many NLP systems is selecting what embeddings to use, we argue that such a step is better left for neural networks to figure out by themselves, i.e., that we should include multiple different types of word embeddings and let the model pick. We examine the usefulness of this idea in the setting of sentence representations (Kiros et al., 2015; Wieting et al., 2015; Hill et al., 2016; Conneau et al., 2017).

The proposed approach turns out to be highly effective, leading to state-of-the-art performance within the same model class on a variety of tasks, opening up new areas for exploration and yielding insights into the usage of word embeddings.

Why Is This a Good Idea? Our technique brings several important benefits to NLP applications. First, it is embedding-agnostic, meaning that one of the main (and perhaps most important) hyperparameters in NLP pipelines is made obsolete. Second, as we will show, it leads to improved performance on a variety of tasks. Third, and perhaps most importantly, it allows us to overcome common pitfalls with current systems:

- **Coverage** One of the main problems with NLP systems is dealing with out-of-vocabulary words: our method increases lexical coverage by allowing to take the union over different embeddings.

- **Multi-domain** Standard word embeddings are often trained on a single domain, such as

Wikipedia or newswire. Our method allows for effectively combining embeddings from different domains, depending on the context.

- **Multi-modality** In many tasks, multi-modal information has proven useful, yet the question of multi-modal fusion remains an open problem. Our method offers a straightforward solution for the question of how to combine information from different modalities.

- **Evaluation** While it is often unclear how to evaluate word embedding performance, our method allows for inspecting the weights that networks assign to different embeddings, providing a direct, task-specific, evaluation method for word embeddings.

- **Interpretability and Linguistic Analysis** Different word embeddings work well on different tasks. This is well-known in the field, but knowing *why* this happens is less well-understood. Our method sheds light on which embeddings are preferred in which linguistic contexts, for different tasks, and allows us to speculate as to why that is the case.

Outline In what follows, we introduce context-attentive embeddings and apply the technique in two state-of-the-art sentence encoders, namely BiLSTM-max (Conneau et al., 2017) and shortcut-stacked sentence encoders (Nie and Bansal, 2017). We evaluate on well-known tasks in the field: natural language inference (SNLI and MultiNLI) and sentiment analysis (SST), and show state-of-the-art performance. Lastly, we perform a variety of small experiments to highlight the general usefulness of our technique and how it can lead to new insights.

2 Context-Attentive Embeddings

Commonly, NLP systems use a single type of word embedding, e.g., word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) or FastText (Bojanowski et al., 2016). We propose giving networks access to multiple types of embeddings, and allowing them to select which embeddings they prefer by assigning each embedding type a weight, depending on the context.

Let $\mathbf{w}$ be a vector representation of the word w and let E be the set of word embedding types: $\mathbf{w}_e$, then, is a word representation of the type $e \in E$ (e.g. $\mathbf{w}_{glove}$ is a GloVe embedding). Given a context vector $\mathbf{h}$, we compute a context-dependent attention mask $\alpha = \langle \alpha_1, \ldots, \alpha_{|E|} \rangle$ over the set of embeddings using:

$$\alpha_w = \mathrm{softmax}(U\ f(V\ [\mathbf{h}, \mathbf{w}_{e_1}, \ldots, \mathbf{w}_{e_{|E|}}])),$$

where $[\cdot]$ is concatenation, $V \in \mathbb{R}^{D \times H}$, $U \in \mathbb{R}^{H \times |E|}$ and f is the hyperbolic tangent. We subsequently apply the attention mask on the concatenated vectors to obtain the context-attended embedding:

$$\mathbf{w}_c = [\alpha_1 \mathbf{w}_{e_1}, \ldots, \alpha_n \mathbf{w}_{e_n}].$$

The embeddings are centered and L2-normalized in order to ensure an equal contribution of each embedding type if the weights are identical. The input embeddings are kept fixed during training.

We compare our method against using word embeddings of a single type, and the natural baseline of naively concatenating the embeddings without any weighting[1].

2.1 Sentence Encoders

We evaluate the usefulness of context-attentive word embeddings in two different state-of-the-art sentence encoders: the BiLSTM-Max model used in InferSent (Conneau et al., 2017) and the recently proposed shortcut-stacked sentence encoder (Nie and Bansal, 2017).

2.1.1 BiLSTM-Max

For a sequence of T words, $\{w^t\}_{t=1,\ldots,T}$ a standard bidirectional LSTM (BiLSTM) computes two sets of T hidden states, one for each direction:

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{\mathrm{LSTM}_t}(\mathbf{w}^1, \ldots, \mathbf{w}^t)$$
$$\overleftarrow{\mathbf{h}_t} = \overleftarrow{\mathrm{LSTM}_t}(\mathbf{w}^t, \ldots, \mathbf{w}^T)$$

For BiLSTM-Max, the hidden states are subsequently concatenated for each timestep to obtain the final hidden states, after which a max-pooling operation is applied over their components to get the final sentence representation:

$$\max(\{[\overrightarrow{\mathbf{h}_t}, \overleftarrow{\mathbf{h}_t}]\}_{t=1,\ldots,T})$$

[1] We also tried an "inner-attention" mechanism where the attention weights are not conditioned on a context vector but instead calculated using just the input embeddings, and forward-predicting the weights using only the context vector (similar to a language model), but found the current approach to work best and be the most interesting.

In our case, we instead use the context-attended embedding $\mathbf{w}_c^t$, calculated using $\mathbf{h}_{t-1}$ as the context for the forward model and $\mathbf{h}_{t+1}$ for the backward model:

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{\mathrm{LSTM}}_t(\mathbf{w}_c^1, \ldots, \mathbf{w}_c^t)$$
$$\overleftarrow{\mathbf{h}_t} = \overleftarrow{\mathrm{LSTM}}_t(\mathbf{w}_c^t, \ldots, \mathbf{w}_c^T)$$

The hidden states are combined in the same way as the standard model, using max-pooling.

2.1.2 Shortcut-Stacked

The shortcut stacked sentence encoder simply consists of multiple stacked BiLSTMs with shortcut connections followed by a max-pooling operation. The input of the i-th stacked BiLSTM layer is given as:

$$\mathbf{x}_t^i = [\mathbf{w}^t, \mathbf{h}_t^1, \ldots, \mathbf{h}_t^{i-1}]$$

which is recursively applied for the number of layers, where $\mathbf{x}_t^1 = \mathbf{w}^t$. This is followed by a max-pooling operation to obtain the final sentence representation.

In our case, the initial input layer is given as $x_t^1 = \mathbf{w}_c^t$ instead, calculated using $\mathbf{h}_{t-1}^1$ as the context for the forward model and $\mathbf{h}_{t-1}^1$ for the backward model, as above.

2.2 Entropy regularization

Even though the embeddings are normalized, the possibility exists that some embedding types get preference due to uneven initialization, in which case embeddings with initial low weights can never recover and remain lowly weighted. We address this through applying orthogonal initialization (Saxe et al., 2013) on the layers of the attention mechanism, and adding the negative entropy as an additional regularization term to the loss, i.e.,

$$\mathcal{L} = \mathcal{L}_{task} + \lambda \sum_{i=1}^{|E|} (\alpha_i \log(\alpha_i))$$

where $\mathcal{L}_{task}$ is the task-specific loss and $\lambda \geq 0$ is a regularization coefficient. Minimizing the negative entropy implies that we pay attention to all embeddings, and only pick a specific one when it's beneficial.

3 Natural Language Inference

Natural language inference, also known as recognizing textual entailment (RTE), is the task of classifying pairs of sentences according to whether they are neutral, entailing or contradictive. Inference about entailment and contradiction is fundamental to understanding natural language, and there are two established datasets to evaluate semantic representations in that setting: SNLI (Bowman et al., 2015) and the more recent MultiNLI (Williams et al., 2017).

The SNLI dataset consists of 570k human-generated English sentence pairs, manually labeled for entailment, contradiction and neutral. The MultiNLI dataset can be seen as an extension of SNLI: it contains 433k sentence pairs, taken from ten different genres (e.g. fiction, government text or spoken telephone conversations), with the same entailment labeling scheme.

3.1 Approach

We train sentence encoders with context-attentive embeddings using two well-known and often-used embedding types: FastText (Mikolov et al., 2018; Bojanowski et al., 2016) and GloVe (Pennington et al., 2014). Specifically, we make use of the 300-dimensional embeddings trained on a similar WebCrawl corpus, and compare three scenarios: when used individually, when naively concatenated or in the context-attentive setting.

We also compare our approach against other models in the same class—in this case, models that encode sentences individually and do not allow attention *across* the two sentences[2]. We include the original works that introduced the respective sentence encoders in the table: InferSent (Conneau et al., 2017) for BiLSTM-Max and Shortcut-Stacked Sentence Encoders (SSSE; Nie and Bansal, 2017).

In addition, we include a setting where we combine not two, but four different embedding types, adding FastText wiki-news embeddings[3], as well as the BOW2 embeddings from Levy and Goldberg (2014a) trained on Wikipedia.

3.2 Results

Table 1 shows the results. We observe that for both sentence encoders, the context-attentive em-

[2]This is a common distinction, see e.g. the SNLI leaderboard at https://nlp.stanford.edu/projects/snli/.

[3]See https://fasttext.cc/

Model	SNLI	MNLI
InferSent (Conneau et al., 2017)	84.5	-
NSE (Munkhdalai and Yu, 2017)	84.6	-
G-TreeLSTM (Choi et al., 2017)	86.0	-
SSSE (Nie and Bansal, 2017)	86.1	73.6
ReSan (Shen et al., 2018)	86.3	-
BiLSTM-Max G	85.4	71.4
BiLSTM-Max F	86.0	72.9
BiLSTM-Max G+F Naive	85.8	73.1
BiLSTM-Max G+F CAE	86.1	73.2
Shortcut-Stacked G	86.6	74.1
Shortcut-Stacked F	86.3	73.6
Shortcut-Stacked G+F Naive	86.3	75.1
Shortcut-Stacked G+F CAE	86.7	75.8
Shortcut-Stacked Many Naive	86.2	75.2
Shortcut-Stacked Many CAE	87.0	75.9

Table 1: Natural language inference results on the SNLI and MultiNLI (Mismatched) tasks. CAE=Context Attentive Embeddings. G=GloVe, F=FastText, Many=multiple different embeddings (see Section 3).

beddings (marked CAE in the table) outperform naive concatenation as well as using the individual embedding types. The shortcut-stacked sentence encoder performs best, and when used with CAE outperforms all other models. We can increase performance even further by using the four different types instead of just GloVe and FastText (marked Many in the table).

Interestingly, which individual embedding performs best depends on the sentence encoder, with FastText working better for BiLSTM-Max and GloVe working better for shortcut-stacked. The results indicate that CAE leads to a "best of both worlds", exploiting the respective strengths of the different embedding types. Naive concatenation performed quite well, but we found CAE to work better: it appears to act as a regularizer over the different embeddings, as we found it to have more consistently good behavior across different hyperparameters as well.

Implementation details The two sentences are represented individually using the sentence encoder. As is standard in the literature, the sentence representations are subsequently combined using $\mathbf{m} = [\mathbf{u}, \mathbf{v}, \mathbf{u} * \mathbf{v}, |\mathbf{u} - \mathbf{v}|]$. We train a two-layer classifier with rectifiers on top of the combined representation. Notice that there is no interaction (e.g., attention) between the representations of $\mathbf{u}$ and $\mathbf{v}$ for this class of model.

We tune the sizes of the LSTM encoders and the fully-connected layers in the classifier (512, 1024 or 2048 dimensions), as well as the regularization coefficient, on the validation set. The learning rate is set to $2e^{-4}$, dropout to 0.1, and we optimize using Adam (Kingma and Ba, 2014). The loss is standard cross-entropy. For MultiNLI, which has no designated validation set, we use the in-domain *matched* set for validation and report results on the out-of-domain *mismatched* set.

Model	SST
Const. Tree LSTM (Tai et al., 2015)	88.0
DMN (Kumar et al., 2016)	88.6
DCG (Looks et al., 2017)	89.4
NSE (Munkhdalai and Yu, 2017)	89.7
BiLSTM-Max G	88.2
BiLSTM-Max F	89.1
BiLSTM-Max G+F Naive	88.6
BiLSTM-Max G+F CAE	89.4
Shortcut-Stacked G	89.3
Shortcut-Stacked F	89.0
Shortcut-Stacked G+F Naive	89.4
Shortcut-Stacked G+F CAE	89.7

Table 2: Sentiment classification results on the binary SST task. For DCG we compare against their best single sentence model (Looks et al., 2017).

4 Sentiment Analysis

To showcase the general applicability of the proposed approach, we also apply it to a case where we have to classify a single sentence, namely, sentiment classification. Sentiment analysis and opinion mining have become important applications for NLP research. We evaluate on the binary SST task (Socher et al., 2013), consisting of 70k sentences with a corresponding binary (positive or negative) sentiment label. Table 2 shows that our sentence encoders, if equipped with CAE and GloVe and FastText embeddings, perform at the state of the art as compared to other single-sentence encoders.

Implementation details We tune the same set of hyperparameters on the validation set as in the previous experiment. We found that it was easier

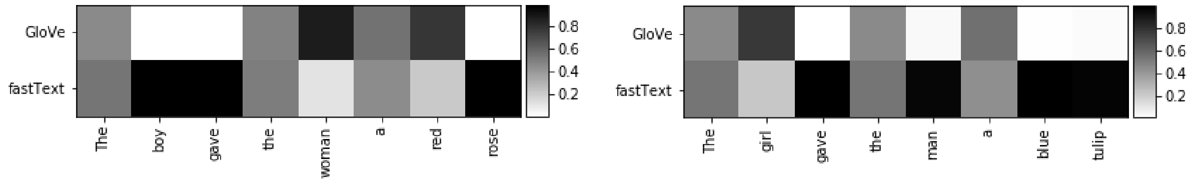

Figure 1: Example visualization of context-attentive embeddings, illustrating the use of different embeddings for opposites (*man*/*woman* and *red*/*blue*).

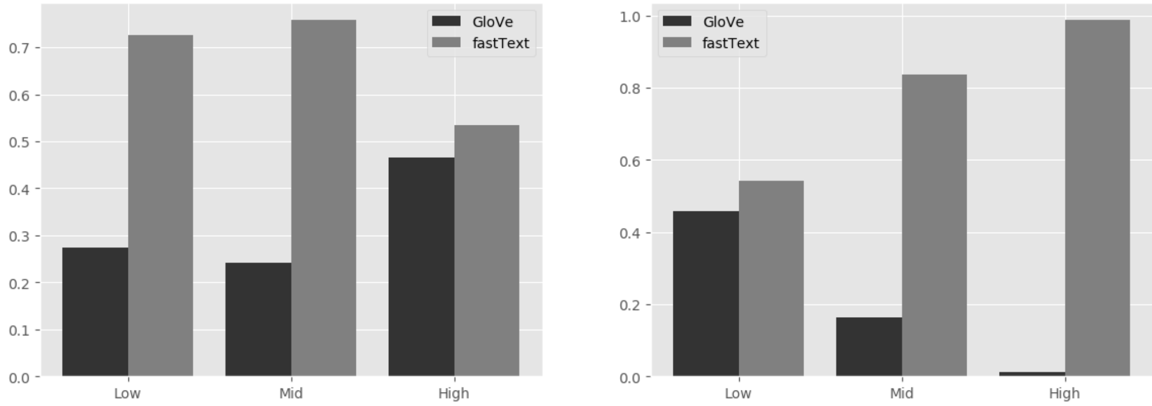

Figure 2: Average attention weight for GloVe and fastText, binned by frequency (left) and concreteness (right).

to overfit with this dataset, given its size, and got better results optimizing with standard SGD with a learning rate of 0.1, shrinking (i.e. multiplying) by 0.2 every time the validation accuracy does not increase.

5 Discussion & Analysis

Aside from improved performance, an additional benefit of context-attentive embeddings is that they allow inspection of the weights that the network assigns to the respective embeddings. In this section, we perform a variety of smaller experiments in order to highlight the usefulness of the technique for studying linguistic phenomena, determining appropriate training domains and evaluating word embeddings.

5.1 Visualizing Contextual Attention

Figure 1 shows the attention weights for GloVe and FastText embeddings, assigned by the best-performing BiLSTM-Max network on the SNLI task. The network has learned to disentangle words that are likely to be close in the same pre-trained embedding spaces but have different meanings (e.g. *girl-boy*, *man-woman* and *red-blue*) by actually using entirely different embeddings for them. We found a similar effect for the sentiment classification models, where opposites are selected from different embedding spaces to facilitate distinguishing between sentiment. See Table 3 for some examples of highest and lowest weighted words for the embedding types.

Word	GloVe	FastText
tall	0.955	0.045
upper	0.939	0.061
woman	0.898	0.102
female	0.873	0.127
short	0.003	0.997
male	0.027	0.973
man	0.032	0.968
lower	0.194	0.806

Table 3: Selected examples from the top weighted words per embedding type.

5.2 Linguistic Analysis

We perform a fine-grained analysis of the behavior of context-attentive embeddings on the validation set of SNLI. Table 4 shows a breakdown of the average attention weights per part of speech, for open versus closed class, and the overall weight assignment. The analysis allows us to make several interesting observations: it appears that FastText embeddings are highly preferred for open

Word type	GloVe	FastText
N	.103	.897
V	.290	.710
JJ	.142	.858
RB	.011	.989
PRP	.130	.870
Other	.574	.426
Open	.154	.846
Closed	.556	.444
Overall	.343	.657

Table 4: Average attention weights per embedding type for different word types, grouped by POS-tag (e.g. N = NN|NNP|NNS, etc.) or open/closed class, contrasted with overall weights, over the SNLI dev set. Underlined are those above the overall average per embedding type.

class words (e.g., nouns, verbs, adjectives and adverbs), while closed class words get more evenly divided attention. Out of the open class words, GloVe does best on verbs.

We can further examine the average attention weights by analyzing them in terms of frequency and concreteness. We use Peter Norvig's English frequency counts from the Google N-grams corpus[4] to divide the words into frequency bins. For concreteness, we use the concreteness ratings from Brysbaert et al. (2014). Figure 2 shows the average attention weights per frequency or concreteness bin, ranging from low to high. We observe that there is a general preference for FastText embeddings across all frequency bins. Interestingly, GloVe gets higher attention weights for abstract words, while FastText is strongly preferred for concrete ones.

5.3 Multi-Domain Embeddings

The MultiNLI dataset consists of various genres. This allows us to inspect the applicability of source domain data for a specific genre. We train embeddings on three kinds of data: Wikipedia, the Toronto Books Corpus (Zhu et al., 2015) and the English OpenSubtitles[5]. We examine the attention weights on the five genres in the in-domain (matched) set, consisting of fiction; transcriptions of spoken telephone conversations; government reports, speeches, letters and press releases; popular culture articles from the Slate Magazine archive; and travel guides.

[4] http://norvig.com/mayzner.html

[5] http://opus.nlpl.eu/OpenSubtitles.php

	Books	Wiki	Subtitles
Fiction	.411	.419	.170
Telephone	.408	.430	.162
Government	.319	.525	.156
Slate	.370	.455	.176
Travel	.357	.476	.167
Overall	.370	.464	.165

Table 5: Average attention weights on different word embedding domains (trained with fastText) with selected MultiNLI domains. Underlined are those above the overall average per embedding type.

Model	Levy	LEAR	SNLI
Only Levy	1.0	-	83.8
Only LEAR	-	1.0	81.7
Levy+LEAR (norm.)	.999	.001	84.2
Levy+LEAR (unnorm.)	.934	.066	83.9

Table 6: Comparison of SGNS (Levy) and LEAR: average attention weights for each, and their corresponding SNLI test accuracy.

Table 5 shows the average attention weights for the three embedding types over the five genres. We observe that Toronto Books, which consists of fiction, is very appropriate for the fiction genre, while Wikipedia is highly preferred for the government genre, where formal language is preferred. Slate and fiction make more use of OpenSubtitles. Surprisingly, the spoken telephone genre does not appear to prefer OpenSubtitles, which we might have expected given that they both consist of dialogue.

5.4 Evaluating Lexical Specialization

Given the recent interest in the community in specializing, retro-fitting and counter-fitting word embeddings for given tasks, we examine whether the lexical-level benefits of specialization extend to sentence-level downstream tasks. After all, one of the main motivations behind work on lexical entailment is that it allows for better downstream textual entailment. Hence, we take the LEAR embeddings by Vulić and Mrkšić (2017), which currently hold the state of the art on the HyperLex lexical entailment evaluation dataset (Vulić et al., 2017). We compare their best-performing embeddings against the original embeddings that were

	SG				CBOW			
	2	5	10	20	2	5	10	20
N	.307	.225	.298	.169	.232	.182	.395	.191
V	.302	.230	.233	.235	.285	.239	.313	.163
JJ	.366	.358	.171	.104	.242	.158	.500	.100
RB	.453	.219	.184	.143	.451	.135	.227	.186
PRP	.337	.232	.272	.160	.414	.207	.209	.170
O	.279	.262	.254	.206	.301	.236	.217	.246
Open	.317	.248	.260	.175	.250	.191	.389	.170
Closed	.281	.260	.254	.204	.306	.235	.217	.243
Overall	.300	.254	.258	.188	.276	.212	.308	.204

Table 7: Average weights per word type for different window sizes. On the left, a comparison for skip-gram (SG); on the right, a comparison for CBOW. Underlined are those above the overall average per embedding type.

used for specialization, derived from the BOW2 embeddings of Levy and Goldberg (2014a). Since LEAR incorporates hierarchical information in the norms of the embeddings, we examine both the normalized and unnormalized case.

Table 6 shows that, unfortunately, lexical specialization does not entail (pun intended) sentence-level improvement. Context-attentive embeddings allow us to inspect the attention weights of each embedding type, and the findings suggest that even though there is a small improvement, the network almost never selects the LEAR embeddings. This experiments illustrates how the proposed approach can be useful for evaluating embeddings, and the question of lexical specialization and how it transfers to downstream tasks is worth further investigation by the community.

5.5 Utilizing Different Contexts Windows

It has been found that the size of a context window when learning embeddings has an impact on their performance (Bullinaria and Levy, 2007; Kiela and Clark, 2014): bigger context windows are likely to lead to more topical embeddings, while smaller contexts are probably more syntactically oriented.

Context-attentive embeddings allow us to examine this question. We train word embeddings on an identical corpus of English Wikipedia using FastText, all with the same hyperparameters except for a different window size, and use them as input for training a BiLSTM-max model on SNLI. Table 7 shows the results on the SNLI dev set, for two BiLSTM-max models trained on that task: one using skip-gram embeddings and one using CBOW embeddings with four different window sizes, including very small ones to relatively big ones: 2, 5, 10 or 20.

We observe differences in the attention over embeddings trained with different window sizes[6]. For instance, both models prefer smaller context windows for adverbs and pronouns; and in both cases the window size of 20 is not used as much for adjectives, adverbs and pronouns. In the case of skip-gram, the distribution for nouns and verbs is pretty even, while for CBOW a window size of 20 is again too big. Overall, skip-gram prefers smaller context windows while CBOW's highest-weighted context-window size is 10. This experiment illustrates how context-attentive embeddings can be used to gain insight into the qualities of different types of embeddings.

6 Related Work

Thanks to their widespread popularity in NLP, a sprawling literature has emerged about learning and applying word embeddings—much too large to fully cover here, so we focus on previous work that combines multiple embeddings for downstream tasks.

Maas et al. (2011) combine unsupervised embeddings with supervised ones for sentiment classification. Yang et al. (2016) and Miyamoto and Cho (2016) learn to combine word-level and character-level embeddings. Contextual represen-

[6]To ensure a meaningful comparison between embeddings of different window sizes, the models were trained with an entropy regularization coefficient of 1.0, which aggressively smooths out the attention distribution. We still see clear differences, which would have even greater with a smaller coefficient.

tations have been used in neural machine translation as well, e.g. for learning contextual word vectors and applying them in other tasks (McCann et al., 2017) or for learning context-dependent representations to solve disambiguation problems in machine translation Choi et al. (2016).

Neural tensor skip-gram models learn to combine word, topic and context embeddings (Liu et al., 2015); context2vec (Melamud et al., 2016) learns a more sophisticated context representation separately from target embeddings; and Li et al. (2016) learn word representations with distributed word representation with multi-contextual mixed embedding. Recent work in "meta-embeddings", which by ensembles embedding sets, has been gaining traction Yin and Schütze (2015); Coates and Bollegala (2018)—here, we show that the idea can be applied in context, and to sentence representations. Peters et al. (2018) recently proposed deep contextualized word representations derived from language models, which led to impressive performance on a variety of tasks.

There has also been work on learning multiple embeddings per word (Chen et al., 2014; Neelakantan et al., 2015; Vu and Parker, 2016), including a lot of work in sense embeddings where the senses of a word have their own individual embeddings (Iacobacci et al., 2015; Qiu et al., 2016), as well as on how to apply such sense embeddings in downstream NLP tasks (Pilehvar et al., 2017).

The question of combining multiple word embeddings is related to multi-modal and multi-view learning. For instance, combining visual features from convolutional neural networks with word embeddings has been examined (Kiela and Bottou, 2014; Lazaridou et al., 2015), see Baltrušaitis et al. (2018) for an overview. There has also been work on unifying multi-view embeddings from different data sources (Luo et al., 2014).

The usefulness of different embeddings as initialization has been explored (Kocmi and Bojar, 2017), and different architectures and hyperparameters have been extensively examined (Levy et al., 2015). Problems with evaluating word embeddings intrinsically are well known (Faruqui et al., 2016), and various alternatives for evaluating word embeddings in downstream tasks have been proposed (e.g., Tsvetkov et al., 2015; Schnabel et al., 2015; Ettinger et al., 2016). For more related work with regard to word embeddings and their evaluation, see Bakarov (2017).

At a higher level, our work draws inspiration from the well-known attention mechanism (Bahdanau et al., 2014), and also relates to self-attention (Lin et al., 2017) and inter-attention (Cheng et al., 2016), where the attention mechanism is applied within the same representation as opposed to learning to align multiple sentences.

7 Conclusion

We argue that the decision of which word embeddings to use in what setting should be left to the neural network. While people usually pick one type of word embeddings for their NLP systems and then stick with it, we find that a context-attentive approach, where embeddings are selected depending on the context, leads to better results. In addition, we showed that the proposed mechanism leads to better interpretability and insightful linguistic analysis. We showed that the network learns to select different embeddings for different data and different tasks.

In future work, we plan to apply this idea to different tasks: it would be interesting to analyze what contexts get picked in multi-modal settings, for example in image-caption retrieval, and to explore what kinds of embeddings are most useful for core NLP tasks, such as tagging, chunking, named entity recognition, and parsing. It would also be interesting to further examine specialization and how it transfers to downstream tasks. In addition, it would be interesting to explore how the attention weights change during training, and if, e.g., annealing the entropy regularization coefficient might help. We limited ourselves to learning representations, but the same idea can be applied to generative settings, such as in language modeling or machine translation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Amir Bakarov. 2017. A survey of word embeddings evaluation methods. *arXiv preprint arXiv:1801.09536*.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of EMNLP*.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911.

John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Heeyoul Choi, Kyunghyun Cho, and Yoshua Bengio. 2016. Context-dependent word representation for neural machine translation. *arXiv preprint arXiv:1607.00578*.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2017. Unsupervised learning of task-specific tree structures with tree-lstms. *arXiv preprint arXiv:1707.02786*.

Stephen Clark. 2015. Vector space models of lexical meaning. *Handbook of Contemporary Semantic Theory, The*, pages 493–522.

Joshua Coates and Danushka Bollegala. 2018. Frustratingly easy meta-embedding—computing meta-embeddings by averaging source word embeddings. In *Proceedings of NAACL-HLT*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of EMNLP*.

Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 134–139.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*.

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *J. Artif. Intell. Res.(JAIR)*, 57:345–420.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of ACL*, volume 1, pages 95–105.

Douwe Kiela and Léon Bottou. 2014. Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 36–45.

Douwe Kiela and Stephen Clark. 2014. A systematic study of semantic vector space model parameters. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 21–30.

Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Proceedings of EMNLP*, pages 2044–2048.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Tom Kocmi and Ondřej Bojar. 2017. An exploration of word embedding initialization in deep-learning tasks. *arXiv preprint arXiv:1711.09160*.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pages 1378–1387.

Angeliki Lazaridou, Nghia The Pham, and Marco Baroni. 2015. Combining language and vision with a multimodal skip-gram model. *arXiv preprint arXiv:1501.02598*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017*.

Omer Levy and Yoav Goldberg. 2014a. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 302–308.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Jianqiang Li, Jing Li, Xianghua Fu, Md Abdul Masud, and Joshua Zhexue Huang. 2016. Learning distributed word representation with multi-contextual mixed embedding. *Knowledge-Based Systems*, 106:220–230.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2015. Learning context-sensitive word embeddings with neural tensor skip-gram model. In *Proceedings of IJCAI*, pages 1284–1290.

Moshe Looks, Marcello Herreshoff, DeLesley Hutchins, and Peter Norvig. 2017. Deep learning with dynamic computation graphs. *arXiv preprint arXiv:1702.02181*.

Yong Luo, Jian Tang, Jun Yan, Chao Xu, and Zheng Chen. 2014. Pre-trained multi-view word embedding using two-side neural network. In *Proceedings of AAAI*.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of ACL*, pages 142–150.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Learning word vectors for 157 languages. In *Proceedings of LREC*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. *arXiv preprint arXiv:1408.6179*.

Yasumasa Miyamoto and Kyunghyun Cho. 2016. Gated word-character recurrent language model. *arXiv preprint arXiv:1606.01700*.

Nikola Mrkšić, Diarmuid O Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of NAACL*.

Tsendsuren Munkhdalai and Hong Yu. 2017. Neural semantic encoders. In *Proceedings of ACL*, volume 1, page 397.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2015. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654*.

Yixin Nie and Mohit Bansal. 2017. Shortcut-stacked sentence encoders for multi-domain inference. *arXiv preprint arXiv:1708.02312*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Mohammad Taher Pilehvar, José Camacho-Collados, Roberto Navigli, and Nigel Collier. 2017. Towards a seamless integration of word senses into downstream NLP applications. *arXiv preprint arXiv:1710.06632*.

Lin Qiu, Kewei Tu, and Yong Yu. 2016. Context-dependent sense embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 183–191.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of EMNLP*, pages 298–307.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. 2018. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. *arXiv preprint arXiv:1801.10296*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of word vector representations by subspace alignment. In *Proceedings of EMNLP*, pages 2049–2054.

Peter D Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

Thuy Vu and D Stott Parker. 2016. k-embeddings: Learning conceptual embeddings for words using context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1262–1267.

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2017. Hyperlex: A large-scale evaluation of graded lexical entailment. *Computational Linguistics*, 43(4):781–835.

Ivan Vulić and Nikola Mrkšić. 2017. Specialising word vectors for lexical entailment. *arXiv preprint arXiv:1710.06371*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. *arXiv preprint arXiv:1607.02789*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Zhilin Yang, Bhuwan Dhingra, Ye Yuan, Junjie Hu, William W. Cohen, and Ruslan Salakhutdinov. 2016. Words or characters? fine-grained gating for reading comprehension. *arXiv preprint arXiv:1611.01724*.

Wenpeng Yin and Hinrich Schütze. 2015. Learning word meta-embeddings by using ensembles of embedding sets. *arXiv preprint arXiv:1508.04257*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of ICCV*, pages 19–27.